UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/750,002 | 12/30/2003 | Hans-Christoph Rohland | 2058.331US1 | 9951 |

50400 7590 03/09/2010

SCHWEGMAN, LUNDBERG & WOESSNER/SAP
P.O. BOX 2938
MINNEAPOLIS, MN 55402

| EXAMINER | |
|---|---|
| ALI, FARHAD | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2446 | |

| NOTIFICATION DATE | DELIVERY MODE |
|---|---|
| 03/09/2010 | ELECTRONIC |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

uspto@slwip.com
request@slwip.com

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 10/750,002 | ROHLAND ET AL. | |
| | Examiner | Art Unit | |
| | FARHAD ALI | 2446 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *09 November 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-9,11-21,23-25 and 27-31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-9,11-21,23-25 and 27-31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *30 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

## DETAILED ACTION

### Status of Claims:

**Claims 1-9, 11-21, 23-25, and 27-31 are pending in this Office Action.**

**Claims 1, 6, 14, 17, and 18 are amended.**

**Claims 10, 22, and 26 are canceled.**

**Claims 29-31 are new.**

### *Claim Rejections - 35 USC § 101*

1. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 6-13 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

The program is embodied on a computer readable storage medium. Said medium is interpreted to include non-statutory subject matter such as carrier waves, signals, and communication media because carrier waves, signals, and communication media store data within the wave, signal or media. While the specification lists examples [on page 8, paragraph [0026]] of physical media, those are considered to be just non-limiting examples. The examiner encourages applicant to amend the claims and specification with explicit arguments that the medium is 'non-transitory' or 'non-transmissible'.

***Specification***

2. The disclosure is objected to because of the following informalities: The specification must provide support for claim amendments as described in regards to rejections under 35 USC § 101 in order to maintain proper antecedent basis for the claims. Appropriate correction is required.

***Claim Rejections - 35 USC § 103***

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Zargham et al. (US 6,954,757 B2) in view of Story et al. (US 6,081,807) and further in view of Kumar et al. (US 7,203,700).

**Claim 1**

Zargham teaches a system comprising:

a database ([Zargham] Column 8 Lines 34-37, "Central repository--refers to a sharable unified capacity such as the operational data store (ODS) with a relational database management system (RDBMS) in the ZLE framework"); and

a plurality of instances of an application server implementing a Java application model ([Zargham] Column 16 Lines 12-16, "The workflow service in the ZLE framework is, for example, an EJB (Enterprise Java Bean, Java 2 enterprise edition ( J2EE)) compliant service running on parallel, available application servers that can store its workflow as XML data structures") coupled in a star topology with the message server at a center of the star topology, the plurality of instances sharing the database ([Zargham] Column 3 Lines 21-26, "the ZLE framework defines a multilevel architecture with a hub, wherein the enterprise applications are loosely coupled to the hub and communicating therewith via adapters" and see Column 6 Lines 53-61, "Loosely coupled applications").

Zarhgham does not disclose a message server having no persistent state such that the message server can be restarted after a failure without performing state recovery operations.

Story et al. teaches a stateless server in Column 1 lines 27-35, "The NFS protocol is defined in various standards documents, e.g., "NFS: Network File System Protocol Specifications," Sun Microsystems, Inc., RFC (Request for Comment) 1094, which is hereby incorporated by reference. The NFS protocol requires a "stateless server." This means that the state of interactions between the server and a client are not to be tracked or managed by the server during a session" in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of Zargham to include the stateless server as taught by Story et al. in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

The modified Zargham does not specifically disclose one or more of the plurality of instances to initiate registering or reregistering of instance-specific information with the message server upon starting or restarting, respectively, of the message server, the instance-specific information including a confirmation of a connection between the one or more of the plurality of instances and the message server.

In an analogous art, Kumar et al. teaches in column 5 lines 1-15, "In act 20 (FIG. 2) connectivity between the new instance J (FIG. 3) and at least one component in a network of computers is established, in any manner, depending on the embodiment. For example, client computers 48 and 49 (FIG. 3) may be informed of the new instance J so that processes therein can start sending work (as illustrated by the dashed lines) to the new instance. Thereafter, as illustrated by act 30 (FIG. 2), the new instance is started, for example by issuing an operating system instruction to the computer in which the instance J is to be started. As noted above, such a computer may be part of a cluster of computers in which all of the other instances are executing" and in column 6 lines 3-9, "Next, in act 53, the starter computer automatically checks that the network has connectivity to the newbie computer, e.g. confirms that one or more processes that implement connectivity of the newbie computer to other processes in other computers of the network are up and running, and are responsive to messages being sent to the newbie computer from the network" in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of the modified Zargham to include Automatic addition of instances as taught by Kumar et al. in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

**Claim 2**

The modified Zargham teaches the system of claim 1 wherein each instance comprises:

a dispatcher node; and a plurality of server nodes ([Zargham] See Figure 9, ZLE framework).

**Claim 3**

The modified Zargham teaches the system of claim 2 wherein each server node comprises:

a java 2 enterprise edition ( J2EE) engine ([Zargham] Column 16 Lines 12-16, "The workflow service in the ZLE framework is, for example, an EJB (Enterprise Java

Bean, Java 2 enterprise edition ( J2EE)) compliant service running on parallel, available application servers that can store its workflow as XML data structures").

**Claim 4**

The modified Zargham teaches the system of claim 1 further comprising:

a central lock server to provide cluster wide locks to the plurality of instances ([Zargham] Column 7 Lines 43-46, "An event may unlock or prompt the commencement of one or more business transactions. An event may lock or prompt the ending of one or more business transactions").

**Claim 5**

The modified Zargham teaches the system of claim 1 wherein the message server comprises:

a first data structure to store a list of connected clients; and a second data structure and a list of services provided in the system ([Zargham] Column 1 Lines 43-46, "the ZLE can integrate data related to the real time operations of the enterprise into a data storage cache, also known as operational data store (ODS)").

**Claim 6**

Zargham teaches a computer readable storage media containing executable computer program instructions which when executed cause a digital processing system to perform a method comprising:

starting a central services node to provide a locking service and a messaging service ([Zargham] Column 7 Lines 43-46, "An event may unlock or prompt the commencement of one or more business transactions. An event may lock or prompt the ending of one or more business transactions" and Column 6 Lines 62-67, "Tightly coupled applications—refers to applications that are not stand-alone and are tightly integrated into the ZLE framework. Tightly integrated functionality—e.g., event capture, data extraction, rules, workflow, message transports and transformations—becomes part of the ZLE core functionality");

starting a plurality of application server instances ([Zargham] Column 16 Lines 12-16, "The workflow service in the ZLE framework is, for example, an EJB (Enterprise Java Bean, Java 2 enterprise edition ( J2EE)) compliant service running on parallel, available application servers that can store its workflow as XML data structures"); and

organizing the application server instances into a cluster having star topology with the central services node at a center of the star topology ([Zargham] Column 3 Lines 21-26, "the ZLE framework defines a multilevel architecture with a hub, wherein the enterprise applications are loosely coupled to the hub and communicating therewith via adapters" and see Column 6 Lines 53-61, "Loosely coupled applications").

Zarhgham does not disclose the messaging service having no persistent state.

Story et al. teaches a stateless server in Column 1 lines 27-35, "The NFS protocol is defined in various standards documents, e.g., "NFS: Network File System Protocol Specifications," Sun Microsystems, Inc., RFC (Request for Comment) 1094, which is hereby incorporated by reference. The NFS protocol requires a "stateless

server." This means that the state of interactions between the server and a client are not to be tracked or managed by the server during a session" in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of Zargham to include the stateless server as taught by Story et al. in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

The modified Zargham does not specifically disclose registering or reregistering instance-specific information with the central services node upon starting or restarting, respectively, of the central services node, the registering or registering initiated by one or more of the plurality of application server instances, the instance-specific information including a confirmation of a connection between the one or more of the plurality of instances and the central services node.

In an analogous art, Kumar et al. teaches in column 5 lines 1-15, "In act 20 (FIG. 2) connectivity between the new instance J (FIG. 3) and at least one component in a network of computers is established, in any manner, depending on the embodiment. For example, client computers 48 and 49 (FIG. 3) may be informed of the new instance J so

that processes therein can start sending work (as illustrated by the dashed lines) to the new instance. Thereafter, as illustrated by act 30 (FIG. 2), the new instance is started, for example by issuing an operating system instruction to the computer in which the instance J is to be started. As noted above, such a computer may be part of a cluster of computers in which all of the other instances are executing" and in column 6 lines 3-9, "Next, in act 53, the starter computer automatically checks that the network has connectivity to the newbie computer, e.g. confirms that one or more processes that implement connectivity of the newbie computer to other processes in other computers of the network are up and running, and are responsive to messages being sent to the newbie computer from the network" in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of the modified Zargham to include Automatic addition of instances as taught by Kumar et al. in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new

instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

**Claim 7**

The modified Zargham teaches the computer readable storage media of claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

sharing a database among the plurality of application server instances ([Zargham] Column 8 Lines 34-37, "Central repository--refers to a sharable unified capacity such as the operational data store (ODS) with a relational database management system (RDBMS) in the ZLE framework").

**Claim 8**

The modified Zargham teaches the computer readable storage media of 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method wherein starting a plurality of application server instances comprises:

starting, for each application server instance of the plurality, a dispatcher node and a plurality of server nodes ([Zargham] See Figure 9, ZLE framework).

**Claim 9**

The modified Zargham teaches the computer readable storage media of claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

starting a message server having no persistent state (See claim 6 rejection).

**Claim 11**

The modified Zargham teaches the computer readable storage media of claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

conducting inter instance communication through the messaging server ([Zargham] Column 21 Lines 53-55, "Messaging functions in the ZLE framework may involve a simple messaging scenario of an EAI-type request-response situation").

**Claim 12**

The modified Zargham teaches the computer readable storage media of claim 9 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

restarting the message server without state recovery responsive to a system failure (see claim 6 rejection).

**Claim 13**

The modified Zargham teaches the computer readable storage media of claim 10 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

notifying all registered instances from the message server when an additional instance joins the cluster ([Zargham] Column 13 Lines 66-68,"The robust message store function supports the EAI platform for ZLE hub-based publish and subscribe operations").

**Claim 14**

Zargham teaches a system comprising:

means for organizing a plurality of application servers instances into a cluster having a star topology with a central services node at a center of the star topology ([Zargham] Column 3 Lines 21-26, "the ZLE framework defines a multilevel architecture with a hub, wherein the enterprise applications are loosely coupled to the hub and communicating therewith via adapters" and see Column 6 Lines 53-61, "Loosely coupled applications");

means for sharing a storage resource across the cluster; and means for performing centralized inter instances communication ([Zargham] Column 8 Lines 34-37, "Central repository--refers to a sharable unified capacity such as the operational data store (ODS) with a relational database management system (RDBMS) in the ZLE framework").

Zargham et al. does not disclose means for performing centralized inter instances communication without maintenance of persistent state information

Story et al. teaches a stateless server in Column 1 lines 27-35, "The NFS protocol is defined in various standards documents, e.g., "NFS: Network File System Protocol Specifications," Sun Microsystems, Inc., RFC (Request for Comment) 1094, which is hereby incorporated by reference. The NFS protocol requires a "stateless server." This means that the state of interactions between the server and a client are not to be tracked or managed by the server during a session" in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of Zargham to include the stateless server as taught by Story et al. in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

The modified Zargham does not specifically disclose the inter instances communication including registering or reregistering instance-specific information with the central services node upon a starting or restarting, respectively, of the central services node, the registering or reregistering initiated by one or more of the plurality of

application server instances, the instance-specific information including a confirmation of a connection between the one or more of the plurality of application server instances and the central services node.

In an analogous art, Kumar et al. teaches in column 5 lines 1-15, "In act 20 (FIG. 2) connectivity between the new instance J (FIG. 3) and at least one component in a network of computers is established, in any manner, depending on the embodiment. For example, client computers 48 and 49 (FIG. 3) may be informed of the new instance J so that processes therein can start sending work (as illustrated by the dashed lines) to the new instance. Thereafter, as illustrated by act 30 (FIG. 2), the new instance is started, for example by issuing an operating system instruction to the computer in which the instance J is to be started. As noted above, such a computer may be part of a cluster of computers in which all of the other instances are executing" and in column 6 lines 3-9, "Next, in act 53, the starter computer automatically checks that the network has connectivity to the newbie computer, e.g. confirms that one or more processes that implement connectivity of the newbie computer to other processes in other computers of the network are up and running, and are responsive to messages being sent to the newbie computer from the network" in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an

application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of the modified Zargham to include Automatic addition of instances as taught by Kumar et al. in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

**Claim 15**

The modified Zargham teaches the system of claim 14 further comprising:

means for centralized locking of a resource within the cluster ([Zargham] Column 7 Lines 43-46, "An event may unlock or prompt the commencement of one or more business transactions. An event may lock or prompt the ending of one or more business transactions").

**Claim 16**

The modified Zargham teaches the system of claim 14 wherein the means for performing comprises:

a message server having no persistent state (See claim 14 rejection).

**Claim 17**

The modified Zargham teaches the system of claim 14 wherein the means for performing comprises:

means for recording services provided in the cluster ([Zargham] Column 13 Lines 66-68,"The robust message store function supports the EAI platform for ZLE hub-based publish and subscribe operations").

**Claim 18**

Zargham teaches a method comprising:

starting a central services node to provide a locking service and a messaging service ([Zargham] Column 7 Lines 43-46, "An event may unlock or prompt the commencement of one or more business transactions. An event may lock or prompt the ending of one or more business transactions" and Column 6 Lines 62-67, "Tightly coupled applications—refers to applications that are not stand-alone and are tightly integrated into the ZLE framework. Tightly integrated functionality—e.g., event capture, data extraction, rules, workflow, message transports and transformations—becomes part of the ZLE core functionality").

starting a plurality of application server instances ([Zargham] Column 16 Lines 12-16, "The workflow service in the ZLE framework is, for example, an EJB (Enterprise Java Bean, Java 2 enterprise edition ( J2EE)) compliant service running on parallel, available application servers that can store its workflow as XML data structures"); and

organizing the application server instances into a cluster having star topology with the central services node at a center of the star topology ([Zargham] Column 3 Lines 21-26, "the ZLE framework defines a multilevel architecture with a hub, wherein the enterprise applications are loosely coupled to the hub and communicating therewith via adapters" and see Column 6 Lines 53-61, "Loosely coupled applications").

Zarhgham does not disclose the messaging service not maintaining a persistent state.

Story et al. teaches a stateless server in Column 1 lines 27-35, "The NFS protocol is defined in various standards documents, e.g., "NFS: Network File System Protocol Specifications," Sun Microsystems, Inc., RFC (Request for Comment) 1094, which is hereby incorporated by reference. The NFS protocol requires a "stateless server." This means that the state of interactions between the server and a client are not to be tracked or managed by the server during a session" in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of Zargham to include the stateless server as taught by Story et al. in order that "if a client makes a request to a server, and after satisfying that request the server fails and is restarted, the server must be able to handle subsequent related requests from the client without needing to access state data that was lost when the server failed" (Column 1 lines 35-39).

The modified Zargham does not specifically disclose registering or reregistering instance-specific information with the central services node upon starting or restarting, respectively, of the central services node, the registering or reregistering initiated by one or more of the plurality of application server instances, the instance-specific information including a confirmation of a connection between one or more of the plurality of application server instances and the central services node.

In an analogous art, Kumar et al. teaches in column 5 lines 1-15, "In act 20 (FIG. 2) connectivity between the new instance J (FIG. 3) and at least one component in a network of computers is established, in any manner, depending on the embodiment. For example, client computers 48 and 49 (FIG. 3) may be informed of the new instance J so that processes therein can start sending work (as illustrated by the dashed lines) to the new instance. Thereafter, as illustrated by act 30 (FIG. 2), the new instance is started, for example by issuing an operating system instruction to the computer in which the instance J is to be started. As noted above, such a computer may be part of a cluster of computers in which all of the other instances are executing" and in column 6 lines 3-9, "Next, in act 53, the starter computer automatically checks that the network has

connectivity to the newbie computer, e.g. confirms that one or more processes that implement connectivity of the newbie computer to other processes in other computers of the network are up and running, and are responsive to messages being sent to the newbie computer from the network" in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

It would have been obvious to one of ordinary skill in the art at the time to create the invention of the modified Zargham to include Automatic addition of instances as taught by Kumar et al. in order that "Automatic addition of a new instance to a group of existing instances as described above in reference to FIG. 2 provides several advantages. Specifically, depending on the embodiment, manual effort in adding a new instance is reduced or even eliminated. Moreover, human errors in a manual process are also reduced or eliminated, thereby to facilitate guaranteed 24x7 uptime. Furthermore, it is no longer necessary to bring down all of the instances of an application, because a new instance can be automatically added on the fly, namely while the other instances are continuing execution" (column 5 lines 16-27).

**Claim 19**

The modified Zargham teaches the method of claim 18 further comprising:

sharing a database among the plurality of application server instances ([Zargham] Column 8 Lines 34-37, "Central repository--refers to a sharable unified capacity such as the operational data store (ODS) with a relational database management system (RDBMS) in the ZLE framework").

**Claim 20**

The modified Zargham teaches the method of claim 18 wherein starting a plurality of application server instances comprises:

starting, for each instance of the plurality, a dispatcher node and a plurality of server nodes ([Zargham] See Figure 9, ZLE framework).

**Claim 21**

The modified Zargham teaches the method of claim 18 wherein starting a central service node comprises:

starting a message server having no persistent state (see claim 18 rejection).

**Claim 23**

The modified Zargham teaches the method of claim 18 further comprising:

conducting inter instance communication through the messaging service ([Zargham] Column 21 Lines 53-55, "Messaging functions in the ZLE framework may involve a simple messaging scenario of an EAI-type request-response situation").

**Claim 24**

The modified Zargham teaches the method of claim 21 further comprising:

restarting the message server without state recovery responsive to a system failure (See claim 18 rejection).

**Claim 25**

The modified Zargham teaches the method of claim 22 wherein organizing further comprises:

notifying all registered instances from the message server when an additional instance joins the cluster ([Zargham] Column 13 Lines 66-68,"The robust message store function supports the EAI platform for ZLE hub-based publish and subscribe operations").

**Claim 27**

The modified Zargham teaches the system of Claim 1, wherein inter-instance communications are conducted through the messaging server ([Zargham] Column 21 Lines 53-55, "Messaging functions in the ZLE framework may involve a simple messaging scenario of an EAI-type request-response situation").

**Claim 28**

The modified Zargham teaches the system of Claim 26, wherein each registered application server instance is notified by the message server when an additional instance registers with the messaging server ([Zargham] Column 13 Lines 66-68, "The robust message store function supports the EAI platform for ZLE hub-based publish and subscribe operations" and Column 14 lines 8-11 "Performing publish and subscribe through the relational database enables the messaging function to leverage the parallelism, partitioning, and built-in manageability of the RDBMS platform).

**Claim 29**

The modified Zargham teaches the system of claim 1, wherein each of the plurality of instances is started using a first instance-specific bootstrap logic, the first instance-specific bootstrap logic synchronized with a second instance-specific bootstrap logic stored in the database ([Kumar et al.] Column 6 lines 25-36, "Thereafter, in act 55, the starter computer automatically sets up resources needed by the new instance in the newbie computer. Examples of such resources include a directory that may be required by the new instance during execution, to write log files, and/or to save data temporarily to disk. The resources that may be set up in act 55 include any resources (such as memory of a minimum size) needed to bootstrap the instance of the application, so that the new instance of the application can come into existence (also called "static configuration"). Such resources may also include state information, and/or initialization

values that may be needed by the new instance" and Column 7 lines 53-60, "In one implementation, a file (also called "initialization parameter" file) that is used for bootstrap configuration identifies two kinds of entries: (a) global entries for use in starting up all instances, and (b) instance-specific entries that may be cloned from another instance for use with the new instance. The initialization parameter file can be private to each computer in the cluster, or alternatively can be shared by all computers in the cluster").

**Claim 30**

The modified Zargham teaches the system of claim 1, wherein a node within the plurality of instances is started using a first node-specific bootstrap logic, the first node-specific bootstrap logic synchronized with a second node-specific bootstrap logic stored in the database ([Kumar et al.] Column 6 lines 25-36, "Thereafter, in act 55, the starter computer automatically sets up resources needed by the new instance in the newbie computer. Examples of such resources include a directory that may be required by the new instance during execution, to write log files, and/or to save data temporarily to disk. The resources that may be set up in act 55 include any resources (such as memory of a minimum size) needed to bootstrap the instance of the application, so that the new instance of the application can come into existence (also called "static configuration"). Such resources may also include state information, and/or initialization values that may be needed by the new instance" and Column 7 lines 53-60, "In one implementation, a file (also called "initialization parameter" file) that is used for bootstrap configuration identifies two kinds of entries: (a) global entries for use in starting up all instances, and

(b) instance-specific entries that may be cloned from another instance for use with the new instance. The initialization parameter file can be private to each computer in the cluster, or alternatively can be shared by all computers in the cluster").

**Claim 31**

The modified Zargham teaches the method of claim 18, wherein the instance-specific information further includes information about a new service that the one or more of the plurality of instances provide ([Kumar et al.] Column 6 lines 25-36, "Thereafter, in act 55, the starter computer automatically sets up resources needed by the new instance in the newbie computer. Examples of such resources include a directory that may be required by the new instance during execution, to write log files, and/or to save data temporarily to disk. The resources that may be set up in act 55 include any resources (such as memory of a minimum size) needed to bootstrap the instance of the application, so that the new instance of the application can come into existence (also called "static configuration"). Such resources may also include state information, and/or initialization values that may be needed by the new instance" and Column 7 lines 53-60, "In one implementation, a file (also called "initialization parameter" file) that is used for bootstrap configuration identifies two kinds of entries: (a) global entries for use in starting up all instances, and (b) instance-specific entries that may be cloned from another instance for use with the new instance. The initialization parameter file can be private to each computer in the cluster, or alternatively can be shared by all computers in the cluster").

***Response to Arguments***

Applicant's arguments filed in regards to claim 1 have been fully considered but they are not persuasive.

The applicant has argued that the entity requesting the subscription is not an application server instance. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., entity requesting the subscription) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Applicant's arguments with respect to added limitations in the Independent claims have been considered but are moot in view of the new ground(s) of rejection.

***Conclusion***

Any inquiry concerning this communication or earlier communications from the examiner should be directed to FARHAD ALI whose telephone number is (571)270-1920. The examiner can normally be reached on Monday thru Friday, 7:30am to 5:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Jeffrey C. Pwu can be reached on (571) 272-6798. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Farhad Ali/
Examiner, Art Unit 2446